The Claims

1. (Currently Amended) A method comprising:

determining ~~an unallocated subset from a plurality of high performance computing (HPC) nodes, each of the unallocated HPC nodes comprising an integrated fabric;~~ an original subset of a plurality of nodes, the original subset comprising nodes currently unallocated to a job, each node in the plurality of nodes comprising a switching fabric integrated to a card and at least two processors integrated to the card;

selecting ~~an HPC job~~ a job from a job queue; and

executing the selected job using at least a portion of the ~~unallocated~~ original subset ~~of nodes~~.

2. (Currently Amended) The method of Claim 1, wherein selecting the ~~HPC~~ job comprises selecting the ~~HPC~~ job from the job queue based on priority, the selected job comprising dimensions not greater than a topology of the ~~unallocated~~ original subset.

3. (Currently Amended) The method of Claim 2, wherein selecting the ~~HPC~~ job from the job queue based on priority comprises:

sorting the job queue based on job priority;

selecting a first ~~HPC~~ job from the sorted job queue;

determining dimensions of the first ~~HPC~~ job with the topology of the ~~unallocated~~ original subset; and

in response to the dimensions of the first ~~HPC~~ job being greater than the topology of the ~~unallocated~~ original subset, selecting a second ~~HPC~~ job from the sorted job queue.

4. (Currently Amended) The method of Claim 2, wherein the dimensions of the first ~~HPC~~ job are based, at least in part, on one or more job parameters and an associated policy.

5. (Currently Amended) The method of Claim 2, further comprising dynamically allocating a job spare from the ~~unallocated~~ original subset based, at least in part, on the dimensions of the ~~HPC~~ job, wherein executing the selected job comprises executing the selected job using the ~~dynamically allocated~~ job spare.

6. (Currently Amended) The method of Claim 1, wherein the plurality of ~~HPC~~ nodes ~~comprise~~ comprises a first plurality and the method further comprises:

determining that dimensions of the selected job are greater than a topology of the first plurality;

selecting one or more ~~HPC~~ nodes from a second plurality of nodes, each of the ~~second HPC~~ nodes in the second plurality of nodes comprising ~~an integrated fabric~~ comprising a switching fabric integrated to a card and at least two processors integrated to the card; and

adding the ~~selected second HPC~~ nodes selected from the second plurality to the ~~unallocated~~ original subset to satisfy the dimensions of the selected job.

7. (Currently Amended) The method of Claim 6, further comprising returning the ~~second HPC~~ nodes selected from the second plurality to the second plurality.

8. (Currently Amended) The method of Claim 1, further comprising;

determining that a second ~~HPC~~ job that was executing on a second subset ~~in~~ of the plurality of ~~HPC~~ nodes has failed;

adding the second subset to the ~~unallocated~~ original subset; and

adding the failed job to the job queue.

9. (Currently Amended) Software embodied in one or more tangible computer-readable media and when executed operable to:

determine an ~~unallocated~~ original subset ~~from a plurality of high performance computing (HPC)~~ of a plurality of nodes, ~~each of the unallocated HPC nodes comprising an integrated fabric~~ the original subset comprising nodes currently unallocated to a job, each node in the plurality of nodes comprising a switching fabric integrated to a card and at least two processors integrated to the card;

select an ~~HPC~~ job from a job queue; and

execute the selected job using at least a portion of the ~~unallocated~~ original subset ~~of nodes~~.

10. (Currently Amended) The software of Claim 9, wherein the software being operable to select the ~~HPC~~ job comprises the software being operable to select the ~~HPC~~ job from the job queue based on priority, the selected job comprising dimensions not greater than a topology of the ~~unallocated~~ original subset.

11. (Currently Amended) The software of Claim 10, wherein the software being operable to select the ~~HPC~~ job from the job queue based on priority comprises the software being operable to:

sort the job queue based on job priority;

select a first ~~HPC~~ job from the sorted job queue;

determine dimensions of the first ~~HPC~~ job with the topology of the ~~unallocated~~ original subset; and

in response to the dimensions of the first ~~HPC~~ job being greater than the topology of the ~~unallocated~~ original subset, select a second ~~HPC~~ job from the sorted job queue.

12. (Currently Amended) The software of Claim 10, wherein the dimensions of the first ~~HPC~~ job are based, at least in part, on one or more job parameters and an associated policy.

13. (Currently Amended) The software of Claim 10, further dynamically allocate a job spare from the ~~unallocated~~ original subset based, at least in part, on the dimensions of the ~~HPC~~ job, wherein the software being operable to execute the selected job comprises the software being operable to execute the selected job using the dynamically allocated job spare.

14. (Currently Amended) The software of Claim 9, wherein the plurality of ~~HPC~~ nodes comprise a first plurality, the software being further operable to:

determine that dimensions of the selected job are greater than a topology of the first plurality;

select one or more ~~HPC~~ nodes from a second plurality, each of the second ~~HPC~~ nodes comprising an integrated fabric; and

add the selected second ~~HPC~~ nodes to the ~~unallocated~~ original subset to satisfy the dimensions of the selected job.

15. (Currently Amended) The software of Claim 14, further operable to return the second ~~HPC~~ nodes to the second plurality.

16. (Currently Amended) The software of Claim 9, further operable to:

determine that a second ~~HPC~~ job that was executing on a second subset in the plurality of ~~HPC~~ nodes has failed;

add the second subset to the ~~unallocated~~ original subset; and

add the failed job to the job queue.

17. (Currently Amended) A system comprising:

a plurality of ~~high-performance computing (HPC)~~ nodes, each node comprising ~~an integrated fabric;~~ a switching fabric integrated to a card and at least two processors integrated to the card; and

a management node operable to:

determine ~~an unallocated subset from the plurality of HPC nodes~~ an original subset of the plurality of nodes, the original subset comprising nodes currently unallocated to a job;

select ~~an HPC~~ a job from a job queue; and

execute the selected job using at least a portion of the ~~unallocated~~ original subset ~~of nodes~~.

18. (Currently Amended) The system of Claim 17, wherein the management node being operable to select the ~~HPC~~ job comprises the management node being operable to select the ~~HPC~~ job from the job queue based on priority, the selected job comprising dimensions not greater than a topology of the ~~unallocated~~ original subset.

19. (Currently Amended) The system of Claim 18, wherein the management node being operable to select the ~~HPC~~ job from the job queue based on priority comprises the management node being operable to:

sort the job queue based on job priority;

select a first ~~HPC~~ job from the sorted job queue;

determine dimensions of the first ~~HPC~~ job with the topology of the ~~unallocated~~ original subset; and

in response to the dimensions of the first ~~HPC~~ job being greater than the topology of the ~~unallocated~~ original subset, select a second ~~HPC~~ job from the sorted job queue.

20. (Currently Amended) The system of Claim 18, wherein the dimensions of the first ~~HPC~~ job are based, at least in part, on one or more job parameters and an associated policy.

21. (Currently Amended) The system of Claim 18, wherein the management node is further operable to dynamically allocate a job spare from the ~~unallocated~~ original subset based, at least in part, on the dimensions of the ~~HPC~~ job, wherein the management node being operable to execute the selected job comprises the management node being operable to execute the selected job using the dynamically allocated job spare.

22. (Currently Amended) The system of Claim 17, wherein the plurality of ~~HPC~~ nodes comprise a first plurality, the management node being further operable to:

determine that dimensions of the selected job are greater than a topology of the first plurality;

select one or more ~~HPC~~ nodes from a second plurality, each of the second ~~HPC~~ nodes comprising ~~an integrated fabric~~ a switching fabric integrated to a card and at least two processors integrated to the card; and

add the selected second ~~HPC~~ nodes to the ~~unallocated~~ original subset to satisfy the dimensions of the selected job.

23. (Currently Amended) The system of Claim 22, wherein the management node is further operable to return the second ~~HPC~~ nodes to the second plurality.

24. (Currently Amended) The system of Claim 17, wherein the management node is further operable to:

determine that a second ~~HPC~~ job that was executing on a second subset in the plurality of ~~HPC~~ nodes has failed;

add the second subset to the ~~unallocated~~ original subset; and

add the failed job to the job queue.

25. (Withdrawn) A method comprising:

selecting a first job in a job queue based on a priority of the first job, the first job having a higher priority than a second job in the job queue and a third job in the job queue;

determining a first shape among a plurality of nodes for execution of the first job;

determining whether a first subset of the plurality nodes collectively capable of accommodating the first shape is unallocated;

if the first subset is unallocated, allocating the first job to the first subset for execution;

if one or more of the nodes in the first subset are allocated, leaving the first job in the job queue and selecting the second job, the second job having a lower priority than the first job and a priority higher than the third job;

determining a second shape among the plurality of nodes for execution of the second job;

determining whether a second subset of the plurality nodes collectively capable of accommodating the second shape is unallocated;

if the second subset is unallocated, allocating the second job to the second subset for execution; and

during execution of the second job, if a third subset of the plurality of nodes becomes unallocated and the third subset together with at least a portion of the second subset are collectively capable of accommodating the first shape:

stopping execution of the second job and returning the second job to the job queue ahead of the third job;

deallocating the second subset; and

allocating the first job to the third subset and the portion of second subset for execution.

26. (Withdrawn) Software embodied in one or more tangible computer-readable media and when executed operable to:

select a first job in a job queue based on a priority of the first job, the first job having a higher priority than a second job in the job queue and a third job in the job queue;

determine a first shape among a plurality of nodes for execution of the first job;

determine whether a first subset of the plurality nodes collectively capable of accommodating the first shape is unallocated;

if the first subset is unallocated, allocate the first job to the first subset for execution;

if one or more of the nodes in the first subset already allocated, leave the first job in the job queue and select the second job, the second job having a lower priority than the first job and a priority higher than the third job;

determine a second shape among the plurality of nodes for execution of the second job;

determine whether a second subset of the plurality nodes collectively capable of accommodating the second shape is unallocated;

if the second subset is unallocated, allocate the second job to the second subset for execution; and

during execution of the second job, if a third subset of the plurality of nodes becomes unallocated and the third subset together with at least a portion of the second subset are collectively capable of accommodating the first shape:

stop execution of the second job and returning the second job to the job queue ahead of the third job;

deallocate the second subset; and

allocate the first job to the third subset and the portion of second subset for execution.